

CM5 Oracle Port Project Plan Thinking Machines Confidential

Cliff Lasser
March 4, 1993

Contents

1 Project Mission	2
2 Justification for Project	3
3 Project Goals	4
4 Justifications for the Goals	6
4.1 OLTP versus Decision Support	6
4.2 Strategic Alliances	7
4.3 Availability	7
4.4 Competitive Performance Analysis	8
4.4.1 OLTP and Decision Support Performance Analysis	8
4.4.2 Availability Performance Analysis	10
4.5 Why Oracle versus CMSQL?	11
5 Oracle Porting Strategy	11
5.1 Oracle and Existing Oracle Ports	11
5.2 Performance of Existing and Anticipated Oracle Ports	13
5.3 CM5 Oracle Porting Strategy	13
5.3.1 Implementation Strategy For Phase 1	14
5.3.2 Implementation Strategy For Phase 2	15
5.3.3 Implementation Strategy For Phase 3	15
6 Deliverables	16
6.1 Phase 1 Deliverables	16
6.2 Phase 2 Deliverables	16
6.3 Phase 3 Deliverables	17
7 Milestones	17
8 Required CM5 Resources	17
9 Availability	19
10 Staffing	19

11 Assumptions, and Dependencies	19
11.1 Assumptions:	19
11.2 Dependencies:	20
11.2.1 CMIX dependencies	20
12 Risks	21
13 Tasks	24
13.1 Phase 1	24
13.2 Phase 2	28
13.3 Phase 3	29
A List of Unix system calls from Oracle	30
B OLTP and Decision Support Performance Analysis	31
B.1 Disclaimer	32
B.2 Description of the table entries	32
B.3 CM5	33
B.4 Sun2000	33
B.5 CM5 with same decision support performance as Sun2000	33
B.6 IBM Mainframe	34
B.7 NCR/Teradata	34
B.8 Teradata (today)	34
B.9 KSR1	34
B.10 nCube	35
B.11 Meiko	35
B.12 CM5-prime (with Viking and large cache)	35
B.13 CM5-prime (with Viking and large cache) with same decision support performance as Sun2000	36

1 Project Mission

"It has become politically incorrect to buy a mainframe."

*John Lewis, explaining why his company, Amdahl Corp.,
is losing money on mainframe computers.*

The purpose of this project is to implement and deliver the best database system for applications consisting of a mixture of decision support and OLTP. The software we will deliver will be the Oracle RDBMS, and the hardware will be a suitably evolved version of the CM5.

We will target databases larger than 100Gbytes, scaling into the terabytes. We will produce the only system capable of scaling both disk storage and the compute power necessary to keep up with the size of the database, something that mainframe owners dream about today. Our system will also be engineered to commercial marketplace requirements: A minimum of 99.9% availability, and 24x7 operations.

2 Justification for Project

The purpose of this project is to port the Oracle RDBMS software to the CM5. We believe that the CM5 will satisfy a need by large database users that is not currently being met, and we believe that we have unique strengths of value to this market that are not likely to be duplicated by other computer vendors.

There appears to be a large number of corporations who wish to do database mining on large online databases. Existing mainframes are sufficient for doing the OLTP aspects of these applications, but they are not capable of simultaneously doing the database mining due their inherent lack of MIPS. The Oracle RDBMS software on the CM5 may be capable of doing both OLTP and database mining on the largest of databases. This would be a unique combination of abilities that could lead to many sales of large systems.

We believe that the Oracle RDBMS on the CM5 is a particularly good solution to the problems of these corporations for the following set of reasons:

- Oracle is a well known company with credibility, and their product is considered to be technically superior
- The Oracle RDBMS is designed to run efficiently on shared disk distributed multiprocessors (such as the CM5), and Oracle is continuing development on parallel software to take advantage of this class of machine
- Thinking Machines is the leader in MPP technology, and has the greatest credibility in this field
- The CM5, with some evolutionary changes in its software and hardware, can provide the best mix of price/performance, reliability and availability, and scalability of any MPP
- Oracle running on the CM5 will be able to solve the scalability problems of current and future large database
- Oracle on the CM5 is an "open system". Our processors and OS are standard technology, and Oracle has the ability to integrate our system into practically any customer environment. From the customer's point of view, there is little that is proprietary, thereby diminishing risk.

Despite these claims, the CM5 is not the only machine out there that might meet the needs of large database owners. In order for TMC to be the leader in this field, we will have to make improvements to both our hardware and software. See below for a competitive analysis and a description of the necessary changes to our system.

The time is ripe to jump into this market. Existing mainframes are "politically incorrect" and do not meet the complete needs of large databases. There is presently no vendor that provides a complete solution. TMC has the opportunity to fill the needs of large database owners.

3 Project Goals

The work on this project has been broken into three phases. Later sections of this document provide justifications for these goals. The general goals of these phases are:

- Phase 1: The primary goal of this phase is to have a demonstration system that would be of interest to some imaginable set of customers in the minimal amount of time. This system will not be shippable. These imagined customers would be primarily interested in “Decision Support” applications, though the system will need to demonstrate its ability to do OLTP and bulk loading of data. We do not expect to have better price/performance or raw performance than our competitors (Meiko, Sun multiprocessors, DB2, Teradata, and KSR) at the end of this phase.
- Phase 2: The primary goal of this phase is to produce a system that can be shipped to customers. Higher rates of performance and robustness will be required. Our target customers will still be those with decision support applications, though we must be working on, and we must have a credible story on how we are going to have sufficiently high availability to do OLTP. We will also have an initial implementation of Oracle’s Parallel Query Decomposition (PQD). We will lay out our vision of what TMC will do for the commercial world and how we intend to accomplish it.
- Phase 3: Our goal for this phase is to deliver on the vision described in Phase 2. Our product will be the best around for a combination of decision support and OLTP for large databases (greater than 100Gbytes). Scalability to extremely large databases (greater than 1 Tbyte) will be demonstrated. We will have hardware and software with a much better price/performance ratio than we have now, a minimum of 99.9% availability, and 24x7 operation capability. PQD will be fully functional and we will demonstrate amazing decision support capability with our scalability of processors.

From a tactical point of view, we would like to have a first sale of our system by the end of 1993. This customer will have to be focussed on decision support since it is unlikely that we will have sufficient availability for OLTP. However, we must be able to make a commitment that we will finish with Phase 3 sometime in 1994.

The specific goals for Phase 1 are:

- Minimum elapsed time to a demonstration and customer shippable system. Aim for end of March.
- Target systems with 32 to 128 nodes and 100 to 300 gigabytes. Our initial sales are most likely to be 32 node machines with 50 gigabytes of SDA for about \$2.5M.
- Don’t work on scalability. We’ll argue that we have it, but we don’t have to demonstrate it.

- A reasonable level of availability for decision support type of applications. We need to pick levels of robustness, fault tolerance (software and hardware), and mean time to recovery that are required for possible customers, and we need to fully understand what are the current levels for these properties of the system so that we can improve them where necessary.
- We will generally not provide additional robustness beyond that already in Oracle. For example, our Shared File System (SharedFS) will not have RAID1 or RAID5 and will instead depend on Oracle's ability to do mirroring of the redo log.
- We will not pick performance levels that are difficult to achieve - only enough to get by. We must talk with the Oracle sales force to get a feel of what is necessary.
- Characterize the performance of the various components of the system
- Minimal system administration automation capabilities
- Determine reasonable target date, specific functionality and reliability goals for Phase 2

The specific goals for Phase 2 are:

- Better performance for decision support. First implementation of "Parallel Query Decomposition" (PQD). Pick important benchmarks and tune system accordingly, though there are no accepted benchmarks for decision support applications.
- Better system administration automation. System should be friendly to administer.
- Better availability. We will identify specific failures that we want the system to survive and what survival means. We must get these issues outlined for the OS group by 2/15 and fully spec'ed by 3/15 to meet OS group schedules.
- Support efforts needed of sales, customer support, training, documentation and other groups will be communicated to those groups.
- Determine reasonable target date, specific functionality and reliability goals for Phase 3

The specific goals for Phase 3 are:

- Rework hardware and software for much better price/performance ratio, a minimum of 99.9% availability, and 24x7 operation capability
- Demonstrate scalability to extremely large databases (greater than 1 Tbyte)
- Demonstrate very high performance for decision support and for OLTP. First determine how we are going to measure performance.

During all these phases, we have the following common goals:

- Get Oracle to interface with us so that they design their software to run optimally on our machine

- Have scalability for those portions of the software that we control. Notify Oracle for portions of Oracle that we discover are not.
- Collect engineering data so that we are able to design the best possible hardware/software configuration for Oracle in the future

4 Justifications for the Goals

4.1 OLTP versus Decision Support

A common belief is that the needs of OLTP and decision support applications are quite different and that we should therefore target one and not the other. I believe this is false and that we need to start thinking and planning now how we are going to accommodate both. While it will be natural to target decision support before OLTP, we must work on the needs of OLTP or we will not have an adequate system for decision support.

The needs of OLTP applications are:

- High availability is most important requirement. Must be better than 99.9%. This translates to better than 10 hours per year of unscheduled downtime, and these hours had best not come in big chunks. These sites are typically 24 x 7 as well and do not tolerate more than 4 hours of scheduled downtime per month.
- High reliability is important and sometimes critical. It is not always critical or else Tandem, Stratus and Sequoia would have taken over the mainframe business and they haven't.
- Industry standard software support. Oracle RDBMS is an example of this software.
- Databases are typically less than 50 Gbytes
- Support for large number of users (hundreds) doing transactions and queries
- The performance bottleneck is disk access time, not MIPS. Existing mainframes do this type of processing well.
- Extremely robust tertiary storage strategy for backup strategies and offlining infrequently accessed data. Coexistence with IBM tape catalogs may be a requirement since they are a standard.

The needs of Decision Support applications are:

- Databases are at least 100 Gbytes, and possibly in the 1 Tbyte range
- The performance bottleneck is a combination of disk access times, transfer rates and MIPS. Existing mainframes do not have the ability to do database mining on hundreds of Gbytes.
- Industry standard software support. Oracle RDBMS is an example of this software.

- High availability is important because this is a commercial production environment, though not as important as for OLTP applications
- High reliability is not always as important as it is for OLTP
- The number of users tends to be small, and each user tends to run queries that consume large amounts of resources

We should recognize that the needs for both classes of applications are very similar. Total MIPS and disk storage may be different from the point of view of conventional machines, but to a scalable MPP like the CM5, they are just configuration choices. In the case of the Oracle RDBMS, even the software is identical. The only real difference between them is the required level of availability and reliability.

Availability and reliability are components of *quality*. The commercial market is going to demand high quality and we are not going to sell machines in this market unless we make a public commitment to providing it and then delivering. Ultimately, we are going to have to provide availability and reliability that will be in the range required by OLTP applications just to satisfy potential decision support applications.

We should expect customers to want to do OLTP and decision support on the same machines. They do them on separate machines today because these systems are not capable of doing both simultaneously. This forces customers to use large numbers of mainframes, with all their complexity, for what are logically single applications. This is why these people are so interested in scalability: They want a single machine that can grow as their needs grow. The fact that we have the technology to solve their problem is very exciting to them, and should be very exciting to us!

4.2 Strategic Alliances

The scalability of the Oracle RDBMS along with our current hardware should lead us to consider a strategic alliance with Sun Microsystems. Sun is able to deliver Oracle systems that grow to a certain size and TMC will be able to deliver systems that start where Sun leaves off. We should try to make the jump from Sun's largest machine to our smallest machine as seamless as possible. We gain by the additional credibility and by having users migrate to our system as their needs grow, and Sun gains by showing that there is a clear and unlimited growth path for the users of their machines. This will become even more important for Sun as competitors like NCR start introducing their own MPP database systems. (Heck, Sun may even want to own us to cover themselves against NCR and IBM by making sure that we stick with SPARC technology!)

4.3 Availability

Availability means different things to different applications. In the case of the phone system, it means that the system can never fail, though individual calls may be dropped or may not complete. However, your phone is never allowed to go dead. In the case of DataParallel applications, availability means that all components of the system are operating flawlessly - there may not be an

errors or a hardware failure or the whole system goes down. The standard defensive strategy for DataParallel applications is checkpointing and restarting in the case of failure.

A common misconception at TMC is that high availability for an RDBMS application such as Oracle is not attainable in the foreseeable future. However, high availability means something very different for this type of application, and there is no inherent reason why the CM5 cannot be a high availability system in this context. In fact, much of the necessary work has already been done.

The Oracle software is designed to cope with the loss of multiple processors. It assumes that the OS will protect it from disk storage failure through techniques such as mirroring. The Oracle RDBMS will detect the failure of processors and will initiate transaction rollback and other cleanup on the remaining processors. The users who were using the failed processors will have to log back in to still functioning processors and resubmit their transactions and queries. The key point is that the loss of some processors or disks is not considered a system failure as long as the remaining components of the system are able to continue functioning.

In the case of the CM5, this means that we need a system with sufficient hardware and software firewalls to prevent failures from cascading. To reach even higher availability, we need automatic isolation of faulty hardware. Finally, in the event of a total system failure, we need fast failure identification, repair or isolation, and fast rebooting.

The CM5 hardware was designed to be highly available in precisely the ways that Oracle needs. The CM5 router can bypass bad spots in the network and disconnect broken nodes. The JTAG network allows fast failure identification. The fact that the CM5 is an MPP with lots of processors makes it more able to survive failure of individual nodes. The majority of the work that now needs to be done to achieve this high availability is software: We need faster and better diagnostics, hardware isolation software, fast rebooting software, RAID software for the disks, OS modifications so that processor failures do not cascade, and so on. All these things are feasible. We just have to decide to make the necessary software investment and then we would have a high availability system.

This project plan assumes that we will make the effort to boost availability above 99.9% for the Oracle RDBMS application. See the section 9 for more specifics on what needs to be done to achieve this level of availability.

4.4 Competitive Performance Analysis

4.4.1 OLTP and Decision Support Performance Analysis

There are several factors that drive performance. They are:

- The total MIPS of all the processors in the system
- The total number of disks, and their individual average access times and transfer rates

- The total disk IO's realizable per second and maximum data transfer rate to processors, without assuming any locality of reference from processors to disks
- Communication rates between processors
- Latency of IO's and communication between processors

See B for a comparison of the CM5 with a variety of machines, including the Sun2000, IBM mainframes, the KSR1, the Meiko Computing Surface, and the nCube. Here are some conclusions we can easily draw from the tables in B:

- Only the CM5 is scalable for decision support applications above 50 Gbytes. MIPS must grow linearly with number of disks and only the CM5 can support enough processors for databases above 50 Gbytes.¹
- The current CM5 is very expensive in terms of Gbytes and MIPS per dollar. There is too much of a difference in the price/performance between the CM5 and the Sun2000 and that might encourage some users to buy multiple Sun2000's rather than a single CM5.
- Mainframes are good only for OLTP up to 50 Gbytes. Even at that size, a mainframe can only get 12% of the disk transfer rates. This means that decision support applications are not a good match for mainframes, as we have discovered by talking to people like Epsilon and Dow Jones. Even a \$200k Sun2000 is faster for scanning a database. It is clear why the existing mainframe business is in such trouble - these machines just do not have the MIPS to do more sophisticated decision support processing.
- The Sun2000 has the best price/performance for strictly OLTP applications.
- Existing multiprocessors and clusters will not have enough MIPS to do decision support on large databases.
- Small hardware enhancements (such as Viking with large cache) would make a *big* difference in price/performance (factor of 2 or 3). Larger systems would also become more feasible. This could lead to a lower list price, or to larger margins.
- Certain aspects of the CM5 do not need performance improvements. The network, for example, probably has sufficient bandwidth for the current and future generation of SPARC processors. Other features, such as reliability and diagnosability, are more important.
- It is not clear what KSR is attempting to accomplish in the commercial marketplace. They are not going to beat a Sun2000 for OLTP, and yet they currently do not have the ability to scale their machine for decision support on much larger databases.
- We do not know enough about our competitors. We know little about their real abilities, and some of them, NCR and Meiko, may be able to outperform us.

¹We did not consider the possibility that a user may have more disks than necessary in order to increase the disk system throughput for the same size database. This strategy would only increase the need for MIPS, which again only the CM5 can provide.

4.4.2 Availability Performance Analysis

The MTBF for the various components of our system are:

- Disks - 500k hours or about 500 months
- Processors - 1m hours or about 1000 months
- SCNs - 1m hours or about 1000 months
- *need to get info on the other components of the system*

So a 200 Gbyte system that is configured for maximum performance decision support would have the following availability:

- the 200 disks will have a failure every 2 months
- the 200 processors and/or SCNs will have a failure every 5 months
- the 32 SCNs will have a failure every 25 months
- *other components of the system?*

It is difficult to determine the repair time for a disk failure without RAID. At the very least it will cost 30 minutes, and at very worst it will be determined between the ratio of disk to tape drives if the entire database has to be reloaded from tape. This cost will be very site configuration dependent, but we believe that with the current Oracle software it will be closer to the later estimate than to the former. The fact that a recovery could easily take several hours, and that there will be around 6 disk failures per year argues very strongly that we should build a RAID system.

There are a couple of strategies we could take with respect to RAID. The simplest is to do RAID5, but without auto healing and hot sparing software. When a drive failed, the operator would have to replace it and manually initiate the healing. This would still only take on the order of a single hour, if not less.

The repair time for a processing node or an SCN is likely to be on the order of an hour if an operator happens to be on site. There is no recovery time when the system comes back up.

If we only protected the disk system with RAID, but still had to do the healing and sparing manually, and assumed that all other hardware failures would bring down the system, then we would have an availability of:

6 disk failures/year + 2 node failures/year = 8 hours or 99.9%

More sophisticated software would do healing and hot sparing automatically, along with keeping the disk system up during these operations, further increasing the availability above 99.99%!

I'm sure that there must be some other hardware parts of the system that I'm not considering.

It is far more likely that availability will be determined by the robustness of our software and the Oracle RDBMS. However, the Oracle RDBMS is designed to survive the failure of a single Oracle instance assuming that its failure does not cascade to other processors. Since it is likely that our OS software and Oracle port will have bugs, it is important that we make our nodes survive software failures on other nodes.

The robustness of our software and the Oracle RDBMS has ramifications for some of the CM5 configurations described earlier: To maximize system availability, it would probably make sense to run disk server software only on the SCNs. If we also run the Oracle RDBMS on the SCN, then a software failure becomes far more likely on the SCN, which in turn would lead to a complete system shutdown.

4.5 Why Oracle versus CMSQL?

Both Oracle and CMSQL have their own advantages and disadvantages, and so we expect them to both be useful for attracting sales.

Oracle has the advantage of being a well known brand name database system and of supporting simultaneous OLTP and decision support. It comes with a salesforce that is dedicated to selling the Oracle software. If we assume that we have the best MPP platform for Oracle, then this sales force will be selling our machines. There are many consultants who have extensive experience with Oracle and are capable of tailoring a CM5 Oracle system so that it integrates well with a specific customer's environment.

CMSQL's primary advantages are performance and its proprietary nature. Though we have yet to measure the performance of either system, CMSQL is likely to outperform Oracle for large queries by integer factors. Because it will only run on CM5s, and we believe that it is likely to outperform RDBMSs on all other machines, we believe we do not need to worry about competition from other machines from a performance point of view. The primary disadvantages of CMSQL with respect to Oracle are that CMSQL has no OLTP capability - the database cannot be updated while queries are in progress, and a lack of existing consultants to sell and service it. We believe that these problems can be overcome, but it is not the purpose of this document to cover this topic in depth.

5 Oracle Porting Strategy

5.1 Oracle and Existing Oracle Ports

Oracle is designed to run in a Unix environment on a single processor, or in a "cluster" multiprocessor environment as long as certain services are provided. Oracle manages all aspects of parallelism in a multiprocessor environment with the help of a "lock manager", and requires a single shared file

system (SharedFS).

When running on a Unix platform, Oracle makes full use of Unix file, network and multiprocess capabilities. Beyond standard file usage for control information, Oracle supports access to raw disk partitions for the actual database for performance purposes. Network capabilities are heavily used to support a client/server model. Multiprocessing is used by Oracle to manage internal tasks such as archiving, talking to multiple users, redo log management, and other things. Multiprocessing is also used to run user applications on the same nodes as the Oracle servers, which reduces networking between clients and servers. Paging is important because running an Oracle server in 32Mbytes is a tight fit, and paging is mandatory for running user applications.

These Unix capabilities are assumed to exist in a cluster environment, such as the Vax cluster. The database files must be accessible to all the processors. Oracle relies on a lock manager common to these processors to coordinate updates to this single database.

There are three known ports of Oracle to MPPs, the most well known being the nCube. This machine supports an OS that is similar to CMOST. Oracle implemented a SharedFS on top of the OS and used message passing to pass data between the computation and disk nodes. I believe that they have not implemented RAID1 or RAID5 and instead rely on the RDBMS to do its own mirroring for the most crucial files, the log files.

Because the nCube OS is not Unix, Oracle had to implement all the standard Unix services, a light-weight threads package and networking through the front end. Any Oracle program, including the Oracle RDBMS kernel, must be ported to the nCube environment. Even though Oracle applications have been designed to be portable, the effort involved is significant so few are able to run on the nCube nodes. This is even more true of end user applications. The result is that most applications must run outside the machine, causing a bottleneck at the Front End.

The lock manager is a special piece of message passing software that runs on a few dedicated processors and is therefore called a "Distributed Lock Manager" (DLM). The lock manager is not able to detect deadlocks, so nCube's and Oracle's claims of large TPC-B benchmark numbers are of questionable value.

Oracle has spent a great deal of time porting the RDBMS to the nCube, and they are not very happy about that. They did not have kind things to say about nCube. They blame the lack of a real OS and the poor reliability of the OS that is there.

The second port of Oracle is to KSR. We understand that the KSR machine is being used as a bunch of distributed processors and that the multiprocessor capabilities of Oracle are not being used. Instead, KSR is writing their own SQL parallelizer to run on top of multiple SQL engines.

This may well be a flawed strategy: By taking over one of the higher layers in the software hierarchy, KSR may well have reduced Oracle's ability to provide support, customization and consulting to the end users. We have no illusions about our ability to provide these services and we want

to take leverage off Oracle's undeniable strength and reputation in this field. At a technical level, it may not be possible to parallelize some queries at the SQL level. Therefore we are not interested in doing this type of port of Oracle to the CM5.

The third port of Oracle is to the Meiko machine. The Meiko is running Unix on each of their Sparc processors. Meiko has also implemented a SharedFS on top of Unix, not within it. We know little else about this port except that because the processors run Unix, the porting effort was relatively small.

5.2 Performance of Existing and Anticipated Oracle Ports

We have no information on this topic yet

5.3 CM5 Oracle Porting Strategy

Oracle has very clearly told us that the right way to port their RDBMS to the CM5 is to run Unix on our processors, and to build a SharedFS and a DLM. Then they say that we should focus primarily on the performance of the SharedFS for decision support. The performance of the DLM will become important when we target OLTP, but that may be off in the future. Oracle's recommendations come from their experience with the nCube and Meiko ports. The Meiko machine is very similar to our's, and the effort to port Oracle to it was much simpler and quicker than to the nCube.

We intend to follow Oracle's recommendations, and will run the RDBMS on top of Unix on the CM nodes (CMIX). For our file system, we will run CMIX on the SCN processors as well, yielding a computing environment on the CM5 that is very similar to what we have at TMC: We will have separate sets of compute and disk servers. The disk servers will also run NFS so all the nodes will be able to open files on them as well as have the swapping and root partitions that are necessary for Unix. Networking to outside the CM will be done through Ethernet gateways.

Beyond the standard Unix facilities, we will have to implement a shared file system that spans most of the disks in the SDA, and a DLM. Both of these services will be provided by a set of dedicated nodes running servers, with the SharedFS running on the SCNs.

The Oracle RDBMS server and other Oracle applications will be compiled with the right magic flags to support a multiprocessor environment, and we will link in our SharedFS and DLM software. Any end user applications that can run on a Unix workstation should be able to run on our nodes as well with just a recompile and link.

An important benefit of going with a Unix strategy is that we can do most of our development on a bank of Sun workstations since it will be compatible with a bunch of CM5 processors running CMIX. We intend to take advantage of this to lessen our dependence on CMIX while it is being

developed.

The implementation of the SharedFS, DLM, and the Ethernet gateway will depend on which phase of the project we are in. For Phase 1, the SharedFS will be server software that runs on the SCNs that will open files on the disks that are local to that SCN. The nodes will communicate with the SharedFS servers through standard Unix sockets. Because the file system will be distributed across multiple SCNs, the nodes will need to have sockets opened to each SCN with a SharedFS server.

The Phase 1 DLM will be implemented in a similar manner to the SharedFS. Some number of nodes will be running DLM server software. Each node running the RDBMS software will open sockets to each of the DLM servers. Because both the SharedFS and the DLM will be implemented on top of standard Unix features, they will both be runnable and testable on a set of Sun workstations.

The user's data will enter our system through the network system, which during Phase 1 will be Ethernet or FDDI gateways running on the CP with standard Unix packet forwarding software. If a single CP does not provide us with sufficient Ethernet bandwidth, then we will use multiple CP's and configure Unix to have multiple Ethernet gateways.

Beyond Phase 1, we will need to consider the needs of specific customers. Oracle provides excellent network interconnectivity software which will allow us to connect to most standard networks (LU6.2, DECNET, TCP/IP, ...). However, if multiple gateways do not solve the needs of a customer, we will have to investigate other higher speed networks such as HIPPI.

The implementation of Phase 1 should be relatively simple, though its performance will probably be limited by that of Unix networking software, which is known to be less than amazing, and by the non-use of the address engines in the SCNs. However, this implementation strategy should allow us to bring up multiprocessor Oracle in the minimum amount of time.

We expect to fix the network and disk performance problems in Phase 2 and 3 with either special purpose device drivers, or with speed hacking of standard Unix capabilities such as NFS and sockets, or with other not yet known software. We have not yet determined the approach we will take. We will also consider using the CMOST "Scalable File System" if it has the capabilities we require and if it can be interfaced to CMIX. If we determine that we need multiple Ethernet gateways and multiple CP's are not appropriate, we may need new hardware and software that supports other networks.

5.3.1 Implementation Strategy For Phase 1

This section describes in general terms the items that need to be implemented for Phase 1. Some of them could happen in parallel:

- Compile and run Oracle on a single Sun workstation (*this has already been done*)
- Implement SharedFS and DLM and test on a bank of Sun workstations

- Run Oracle on a collection of Sun workstations with the SharedFS and DLM
- Run Oracle on a single node in the CM5 under CMIX, with the database and swapping partitions on the CP
- Run Oracle on a single node, with the database and swapping partitions on an SCN that is also running CMIX
- Run single processor RDBMS's on several nodes in the CM5 using multiple SCNs under CMIX
- Test the SharedFS and DLM on several nodes and SCNs running CMIX
- Run multiprocessor Oracle on multiple nodes with the SharedFS and DLM under CMIX

5.3.2 Implementation Strategy For Phase 2

We will probably have to implement the software listed below. One of the tasks in Phase 1 is to determine the list of tasks for Phase 2 and to determine the performance levels we will attempt to achieve. The items below are an educated guess at what we will have to do.

- Partially fault tolerant DLM. The DLM needs to be able to recover from the failure of a single processor.
- Configuration tools/documentation for CMIX and Oracle.
- Work with Oracle to make PQD as good as possible (we can start on this immediately)
- Implement faster message passing under CMIX for faster IO, PQD and DLM
- Implement and tune disk system. Design and implement simple RAID5 scheme.
- Provide higher bandwidth connections to the outside environment, such as multiple Ethernet gateways.

5.3.3 Implementation Strategy For Phase 3

The goal of Phase 3 is to do high performance and availability decision support and OLTP. Our understanding of what this entails is limited, but we will have to do at least the things mentioned below. One of the tasks in Phase 2 is to determine the list of tasks for Phase 3 and to determine the performance levels we will attempt to achieve. The items below are an educated guess at what we will have to do.

- Greatly increase the robustness and survivability of CMIX in the face of hardware and software failures. Provide "firewalls" between processors so that a single processor failure does not cascade to other processors.
- Improve diagnostic capability of the machine. We need to diagnose failures as quickly as possible to minimize downtime. Diagnostics must be interpretable by normal humans.

- Improve rebooting time. This will help minimize down time.
- Work on various hardware improvements to reduce probability of single point catastrophic failures (such as the power supplies).
- Improve performance of the machine through the use of 40 Mh Vikings with large caches.
- Improve on administration tools, particularly for RAID5 disk system.
- Work with Training, CSG, Documentation and other groups so that TMC can support customers

6 Deliverables

6.1 Phase 1 Deliverables

- Oracle port to network of workstations and to CM5 processors
- Benchmarks that measure performance of initial implementation of Oracle on the CM5
- Minimal administration capabilities and simplest implementation of these capabilities: configuration, startup, shutdown, backup, restore, recovery)
- Minimally working CMIX: CMIX needs to work as well as a bank of Sun workstations. Performance is not important.
- Define functionality, performance and reliability tasks for Phase 2

6.2 Phase 2 Deliverables

- Customer-shippable port of Oracle
- Tests that demonstrate correctness of Oracle port
- Benchmarks that demonstrate good performance of Oracle on the CM5
- Oracle PQD working
- Customer-acceptable administration capabilities
- Minimal RAID5 disk system, with manual heal and sparing utilities
- Other items for higher availability, to be determined
- Really working CMIX
- Tertiary backup mechanism (ITS)
- Define functionality, performance and reliability tasks for Phase 3
- Documentation and customer support training materials
- Sales material describing why Oracle on the CM5 is the database system to buy

6.3 Phase 3 Deliverables

- New and improved CM5 hardware and related software for higher availability, 24x7 operation, and better performance
- Better software for minimizing downtime when it happens
- Better software to prevent cascading of single processor failures
- Better sales material
- Even better performance for OLTP and decision support
- Multiple ethernet connections for transferring large amounts of data, or for supporting large number of users

7 Milestones

For Phase 1, the milestones are:

1. Run Oracle with DFS, in exclusive and parallel mode on a single Sparcstation with DLM but no SharedFS.
2. Run Oracle with DFS, in exclusive and parallel mode on multiple Sparcstations with DLM and SharedFS.
3. Run Oracle with DFS, in exclusive and parallel mode on multiple CMIX nodes, with DLM and SharedFS.
4. Completion of all Phase 1 tasks

There are no milestones for Phase 2 and Phase 3 since we do not yet have a task list.

8 Required CM5 Resources

Here is what we will need for the next few months:

Immediately

- 4 nodes
- 2 SCNs
- 4 disks (2 per scn)

We need this hardware to bring up CMIX in the mode that we plan to use it, and to bring up the initial version of Oracle on multiple processors. Not only will we run Oracle on it, but we will also run our editors and compilers on it, and we will use the SDA as our file server. This will begin the shakeout of CMIX.

Four nodes for Oracle is the bare minimum. We need 2 nodes for Oracle instances, one for DLM, and 1 for archiving. We may be able to get the DLM to run on the same nodes as the Oracle instances.

We need 2 SCNs for testing our SharedFS. Each SCN needs 2 disks so that the server code can be tested too.

April

- 8 nodes
- 3 SCNs
- 6 disks (2 per scn)

This size machine will be necessary to debug all the parallelism that exists in Oracle. We will be able to do some very limited performance analysis.

May

- 16 nodes
- 3 SCNs
- 24 disks (8 per scn)

At this point we need a system that can be set up for customer evaluation, both so that we can start the sales process and so that we can evaluate performance and do some benchmarks.

May and beyond

We will require periodic access to much larger CM5's for performance and customer evaluation. In order to run the TPC-B benchmark, for example, we would need around 512 nodes and 512 disks. Since we would expect to share hardware with CMOST users, it will be especially important to be able to reboot the machine with either CMOST or CMIX and without having to reformat the disks.

Hardware for Oracle's internal development

Oracle is presently using nCubes for all their MPP development. It would be very much in our interest for Oracle to do their development on our machine instead. The software that would come

out would certainly be better tuned to our hardware and software. We would always have the earliest release of new Oracle features, and we would have an inside connection from which we would learn what kind of hardware we really should be building.

The Oracle developers told us that they presently have a 512 node nCube machine. They typically partition it for development into 64 node chunks, and put bigger pieces together for benchmarking. My feeling is that the smallest CM5 that would replace their nCube would be:

- 128 nodes
- 128 disks

We should consider this to be a sales opportunity rather than a giveaway. It seems to me that Oracle should want our machine, assuming we do things correctly. Of course, they will not consider our machine until we have demonstrated that we can do everything we have planned for in phases 1 and 2.

9 Availability

This section is supposed to describe what we need to do to increase availability. To be filled in.

10 Staffing

The staffing of this project consists of Mike Best, Cliff Lasser, Craig Stanfill, Steve Swartz, and Ephraim Vishniac. The task list indicates who is working on which tasks for Phase 1.

We appear to have enough resources to do Phase 1 of this project. However, we will probably require additional people in the future to do things like performance evaluation, sales support, ...

11 Assumptions, and Dependencies

This section lists all the assumptions, dependencies, constraints and risks that we can think of.

11.1 Assumptions:

- (Phase 1 and 2) All our work and what gets shipped to customers will be existing CM5 hardware. We may need 128 MBytes per processor and 32Mbytes per SCN, but at the moment, we are not certain that we will need this additional memory.
- (Phase 3) The improvements in hardware and software for higher availability and performance are implemented

- Quality of Oracle software is good
- (Phase 1 and 2) CMIX is presently being built on top of SUNOS 4.1 and that is what we'll use. We do not care whether it is built on top of either SUNOS 4.1 or Solaris. All we care about is that it happens.
- (Phase 3) CMIX will have to be built on top of Solaris in this time frame because SunOS 4.1 will be obsolete by then

11.2 Dependencies:

- CMIX has to work. See below for more detail.
- Secondary and tertiary storage needs to be implemented (including RAID5). See Craig's "Oracle IO Requirements" document.
- Multiple ethernet connections. This may require additional hardware design and implementation so that we can plug SCSI to ethernet adaptors into the SCN.
- Technical help from Oracle for the porting process
- CM5 resources we've asked for

11.2.1 CMIX dependencies

Clearly, the biggest dependency for bringing up Oracle on the CM5 is CMIX. This section lists all the features and capabilities we expect from CMIX. Some of these items are mentioned elsewhere, but the purpose of this section is to list them all in one place so that there can be no confusion about what is required from CMIX. The phases by which particular capabilities are needed are in parentheses. See Appendix A for an comprehensive list of Unix functions that are called by Oracle.

By the way, CMIX is also a dependency for the ITS and HIPPI projects. It currently appears that the majority of work that needs to be done for CMIX is the same for the Oracle, ITS and HIPPI projects. We believe that once CMIX works with the features listed below, we will need a minimal amount of additional OS group support. All this means is that the ITS and HIPPI projects will require all the same support that Oracle will require.

- (Phase 1) SunOS 4.1 on the nodes and the SCNs. The Oracle project has no special requirement for CMIX to be based on SunOS 4.1 or Solaris. We want the same OS that the rest of TMC is running.
- (Phase 1) The SCN must mount the disks in the standard Unix way. Some subset of the SCNs in a machine will have to support NFS filesystems and swapping partitions for the nodes.
- (Phase 1) CMIX and CMOST must coexist in the following sense: It must be possible to run CMIX and CMOST in different partitions of the same machine. And, a single partition must be bootable with either CMIX or CMOST without causing the disks to be erased. This is

important because we will have to share large CM's with other groups but we don't want our disk files deleted everytime we give up a partition to a CMOST user.

- (Phase 1) We must be able to run diagnostics on a partition running CMIX. We must be able to shut down CMIX and run diagnostics without disturbing the contents of the disks.
- (Phase 1) We need to have a tape system that can be accessed through standard Unix facilities that will allow us to back up individual disks. We do not yet understand how fast backups need to be, but it would be a good guess that we will need multiple tape drives.
- (Phase 2) We need to boot individual processors without having to bring down the whole system. This will be important for minimizing complete system shutdown and therefore maximizing system availability.
- (Phase 2) Kernel driver software to more efficiently do the SharedFS and to take advantage of the address engines in the SCNs. We will determine the needed performance during Phase 1.
- (Phase 2) We need to support multiple Ethernet gateways for performance and reliability. We will determine the needed performance during Phase 1.
- (Phase 3) Kernel driver software to more efficiently do message passing between the nodes. We will determine the needed performance during Phase 1.
- (Phase 3) We will need support for increasing Oracle availability. We will probably need extra firewalling between processors, and the ability to run some subset of the hardware diagnostics without shutting down the entire partition.
- (Phase 3+) We will need some sort of ability for nodes in CMOST partitions to communicate with nodes in the CMIX partition. It has been suggested that users may wish to run Oracle in one partition feeding data to a statistical package, for example, running under CMOST in another partition. We do not yet understand the desired performance. It may be that simply communicating data through files on the CP would be adequate. It is also not clear that this is going to be an issue because the new OS work may solve this problem.

12 Risks

All of these risks apply to all the development phases:

- Risk: Software doesn't scale the way we thought it should. Both the OS and the Oracle application may not scale appropriately.
Solution: If Oracle does not scale, we're hosed. However, we have reason to believe that this is not a real risk since Oracle has been demonstrated to scale on an nCube. There are no obvious reasons for why the OS should not scale, so this may not be a significant risk either.

- Risk: Inadequate testing of underlying software (primarily CMIX)
Solution: Start using CMIX for everyday use. Get some developers to move over to CMIX. Development work of CMIX kernel may conflict with this and we may need a second development platform. Note that CMIX is a required component of several other TMC products: ITS and HIPPI. We need to work with those folks to make sure that CMIX will be sufficiently robust.
- Risk: The CM hardware and/or software is fundamentally unable to meet the requirements for things like frequency of failures, recovery time, maintenance time and undetected errors.
Solution: We go out of business or
We start doing fault tolerant work for identifiable pieces of the system that fail frequently. For example: RAID1 or RAID5, multiple CP network bridges. Make it possible to shutdown parts of the system without bringing everything down. Use network protocols to survive component failures and removal. Make the DLM fault tolerant.
Fast reboot times will be important.
Need to be able to run some sort of diagnostics during the day. Assume that system cannot be brought down at night for diagnostics, and anyway, we want to discover problems sooner than that (if possible).
- Risk: There is immaturity in the CM hardware and/or software in the areas of frequency of failures, recovery time, maintenance time and undetected errors but these things may not be on the main company priority list and so they don't get worked on.
Solution: Oracle needs to be on the main company priority list
- Risk: Network performance is inadequate and unreliable.
Solution: Try to build a model of what our network usage will look like. Do software simulator that places this type of load on real hardware and measure the actual performance. Do this ASAP so that we don't get screwed down the line and have to rethink basic workings of CMIX and the NI.
Get Bromley involved as soon as we have an idea of what the load on the DR is going to look like.
Quite frankly, I don't see getting the resources to do anything about this.
- Risk: There are unknown performance problems with CMIX or other aspects of our system. Our current understanding of Oracle performance is small.
Solution: Start trying to get a handle on this ASAP.
- Risk: We don't get the hardware we need
Solution: To begin with, we need very minimal amounts of hardware for Phase 1, and we already have most of that hardware. For Phase 2, we can probably share the hardware we have requested, but that will make debugging larger systems and doing performance tuning much slower. We would have to stretch out our schedule in this case.
- Risk: Connecting to IBM world is harder than we think. There is a number of ways that data could get into the system. One of them is through transactions, and we should have no problems with that. The other is bulk loads. If the amount of data to be bulk loaded is

particularly large, we may have a couple of problems. The first one is that Oracle does not yet have a parallel bulk loader. This will limit how much data can be loaded into the system. When Oracle has a parallel bulk loader, we could run into another problem: How do we get the data into files that can be read by the parallel bulk loader?

Solution: We could ignore this problem for now since we can't do anything till we have a parallel bulk loader. We are not going to implement our own.

When we get the parallel bulk loader, a solution to the file transfer problem may just simple be to transfer portions of the file through the multiple ethernet connections. We would have to have sufficient total ethernet bandwidth to transfer all the data in a reasonable amount of time.

So this is probably not much of a risk, but until we understand a customer's needs better, it will be difficult to say exactly what the solution will be. There is still the risk that we may have to something that we are not presently planning to do.

- Risk: Oracle's Operating System Dependent (OSD) layer is not as clean as we thought it is - port is significantly more difficult than we thought
Solution: We get more help from Oracle than we are presently planning to receive. We can also stretch out the schedule. We have no indications at this time that this is a problem.
- Risk: Porting kit is either incomplete or incorrect
Solution: Get Oracle to send us what's missing or to fix what's broken. We have no indications at this time that this is a problem.
- Risk: Scalability problems with SunOS with respect to number of file descriptors and sockets
Solution: Recompile kernel with larger parameters. We are almost certainly going to have to do this.

13 Tasks

13.1 Phase 1

- Design SharedFS

Description: The SharedFS will support a very restricted capability file system for storing the Oracle Database. It will allow multiple Oracle servers to open very large files that span multiple disks. Each of these servers will get a read consistent view of the files that they have opened in common. This software is currently designed to run on top of SunOS so that it can run work on a bunch of Sparcstations.

Who: Steve
Time: Done

- Implement SharedFS

Who: Steve
Time: 5 days

- Test SharedFS

Who: Steve
Time: 5 days

- Design message passing (MP) lib for DLM

Description: The MP lib's purpose at the moment is solely to support the nCube DLM. This DLM depends on a variety of nCube message passing functions, and so we are designing our library to be as close as possible to it. This software is currently designed to run on top of SunOS so that it can run work on a bunch of Sparcstations.

Who: Steve and Mike
Time: Done

- Implement MP lib for DLM

Who: Steve
Time: Done

- Test MP lib for DLM

Who: Steve
Time: 5 days

- Implement DLM with MP

Description: Use the nCube DLM, but make minor changes so that we can plug in our MP lib rather than nCube message passing functions.

Who: Mike

Time: Done

- Test DLM without plugging it into Oracle

Description: Make sure that asynchronous events work properly. Do not worry about correctness of the locking process because we have theoretically been given a piece of correct code. We only need to test our interface to the MP lib.

Who: Mike

Time: 2 weeks

- Bring up CMIX on nodes and SCN

Description: Make SunOS run on the nodes and the SCN in diskless client mode. The disk server will be the CP. Just demonstrate that this works. Robustness is not part of this task.

Who: Steve

Time: Done

- Get paging and file system partitions built and mounted on SCN disks

Description: With SunOS running on the SCN, mount the disks on that SCN. Build a filesystem on those disks. Make the nodes mount their paging partitions on the SCN's disks.

Who: Steve

Time: Estimate - 10 days left

- Make CMIX sufficiently robust for running Oracle demos

Description: CMIX needs to be able to run for several hours at a time without crashing. CMIX must not crash while doing common Oracle servers functions. There will be a more detailed list of specific action items once CMIX is running on the nodes and the SCN's.

Who: Steve

Time: Estimate - 6 weeks

- Make CMIX run on CM5 nodes with DASH

Description: The current version of CMIX only runs on older style CM5 boards with MMU chips instead of DASH chips. Those boards presently only have up to 8 Mbytes of memory. Since we need more memory than that, we need to port CMIX to run on nodes with Dash chips. If we can upgrade the

existing MMU boards to have 32Mbytes, then this task goes away.

Who: Bruce
Time: Estimate - 4 weeks

- Compile and run Oracle for a Sparcstation

Description: Compile the Oracle porting kit for a Sparcstation, then run some test cases on a Sparcstation to verify that it was correctly built. Only bring up the Oracle Server, not all the other Oracle tools and language preprocessors.

Who: Cliff
Time: Done.

- Run Oracle on a single CMIX node with the database on the SCN

Description: Run the same binaries that ran on a Sparcstation on a single CMIX node. The Oracle database files should be on a single SCN's disks.

Who: Ephraim
Time: 1 day

- Run Oracle with DFS, in exclusive and parallel mode on a single Sparcstation with DLM but no SharedFS.

Who: Ephraim and Mike
Time: Estimate - 4 weeks

- Run Oracle with DFS, in exclusive and parallel mode on multiple Sparcstations with DLM and SharedFS.

Who: Ephraim and Mike
Time: Estimate - 2 weeks

- Run Oracle with DFS, in exclusive and parallel mode on multiple CMIX nodes, with DLM and SharedFS.

Who: Ephraim and Mike
Time: Estimate - 1 week

- Compile and run all Oracle tools and language preprocessors on a Sparcstation

Description: Go through all components of the Oracle porting kit, compiling then running each one on a Sparcstation.

Who: Ephraim
Time: Estimate - 1 week

- Run existing Oracle tests for all Oracle tools and language preprocessors

on a Sparcstation

Description: Run all Oracle provided tests on all Oracle tools and language preprocessors that were provided to us on the porting kit tapes

Who: Ephraim

Time: Estimate - 2 weeks

- Run existing Oracle tests for all Oracle tools and language preprocessors on multiple Sparcstations

Who: Ephraim

Time: Estimate - 1 week

- Run existing Oracle tests for all Oracle tools and language preprocessors on multiple CM5 nodes

Who: Ephraim

Time: Estimate - 1 week

- Do performance analysis of Oracle on CMIX nodes

Description: This analysis is likely to be very shallow since various parts of the system are likely to be orders of magnitude slower than they should be. It is not at all clear how much analysis we can do in that type of environment other than to verify our existing suspicions of what needs to be sped up (disk and message passing).

Who: Craig, Mike, and Ephraim

Time: Estimate - 3 weeks

- Do I/O system requirements document

Description: Document everything that Oracle will be expecting of the CM5 I/O system. This includes but is not limited to disk, backups, and networking. Document required functionality, performance and reliability of each of this components of the I/O system.

Who: Craig

Time: Done. This doc will certainly need updating

- Decide on faster interprocess communication semantics

Description: Assuming that having the fast interprocess communication is important for PQD and the DLM, come up with communication semantics that are appropriate for PQD and the DLM and whose implementation can be made fast

Who: Steve

Time: Done, but need to write down what has been determined

- Decide on faster disk system strategy

Description: Assuming that having the fastest possible disk system is necessary, come up with the right way to achieve it or something close.

Who: Ephraim

Time: Done.

- Determine Phase 2 performance goals

Who: Cliff

- Determine Phase 2 functionality goals

Who: Cliff

- Produce Phase 2 task list

Who: Cliff

- Produce Phase 2 schedule

Who: Cliff

13.2 Phase 2

- Determine Phase 3 performance goals

Who: Cliff

- Determine Phase 3 functionality goals

Who: Cliff

- Produce Phase 3 task list

Who: Cliff

- Produce Phase 3 schedule

Who: Cliff

- Do various Phase 2 tasks

- Produce sales information about Oracle on the CM5 for sales team

- Produce documentation for running Oracle on the CM5.

Description: This documentation must cover the needs of users, installers, training, and customer support

- Bring up PQD

- Do work determined by I/O system requirements document
- Do work required for faster interprocess communication
- Do work determined by faster disk system strategy
- DLM must tolerate single processor failures.
- Configuration tools/documentation for CMIX and Oracle
- Do performance analysis

13.3 Phase 3

- Do Phase 3 tasks

A List of Unix system calls from Oracle

Oracle invokes all the Unix functions listed below on a Sparcstation under SunOS:

accept	fork	longjmp	setpgid
access	fprintf	longjmp	setpgrp
alarm	fputc	lseek	setsockopt
atoi	fputs	lstat	setuid
atoi	fread	malloc	shmat
atol	free	memchr	shmctl
bcopy	fseek	memcmp	shmdt
bind	fstat	memcpy	shmget
bsearch	ftell	memset	sigblock
builtinalloca	ftok	mkdir	signal
bzero	fwrite	open	sigsetmask
calloc	getcwd	perror	sigvec
chdir	getenv	pipe	sleep
chmod	geteuid	printf	socket
close	getgid	putenv	sscanf
connect	getgrent	puts	stat
creat	getgrnam	qsort	stty
ctime	gethostbyaddr	rand	system
dtou	gethostbyname	read	time
dup	gethostname	readv	times
dup2	getpid	realloc	ttyname
econvert	getppid	recv	ulimit
endgrent	getpwnam	recvfrom	umask
endpwent	getpwuid	remove	uname
execl	getrusage	rename	ungetc
execve	gets	rewind	unlink
execvp	getservbyname	scandir	usleep
exit	getsockname	select	vfork
exit	gettimeofday	semctl	vfprintf
fabs	getuid	semget	vprintf
fclose	gtty	semop	vsprintf
fcntl	index	send	wait
fdopen	inetaddr	sendto	write
fflush	ioctl	setbuf	
fgets	isatty	setgid	
filbuf	kill	setgroups	
flock	link	setitimer	
flsbuf	listen	setjmp	
fopen	localtime	setjmp	

B OLTP and Decision Support Performance Analysis

The type of database use, OLTP or Decision Support, will stress different aspects of the performance of a system. OLTP use will tend to stress total disk IO's per second and low latency. Decision Support will tend to stress total MIPS, and maximum transfer and communication rates. This section will attempt to characterize the performance of various CM5 configurations versus other machines.

There are several parameters that apply across all machines that can run Oracle:

- The number of Mbytes of data that a single processor can scan is a function of its MIPS and the efficiency of its primary memory system. Here some specific numbers:
 - 0.5 Mbytes/sec for a Cypress Sparc (currently the CPU in our nodes and SCNs) ²
 - 1.5 Mbytes/sec for a 40Mh Viking with a large secondary cache ³
 - 1.0 Mbytes/sec for a Sun2000 CPU in a 20 CPU system ⁴
 - 1.0 Mbytes/sec for an IBM mainframe CPU ⁵
- The disk performance is assumed to be the same for all machines
 - All disks are assumed to be 1 Gbyte disks. In 1994 this will become 2 Gbyte per disk.
 - Sustained transfer rate is assumed to be 1.0 Mbyte per sec per disk ⁶
 - Sustained number of IO's is 50 per sec per disk
 - SCSI channels can sustain at most 10 disks at those rates
- Block size is assumed to be 2k bytes for OLTP systems and 8k bytes for Decision Support ⁷

An interesting fact to note about the CM5 is that we have Sparc processors in our disk nodes (SCNs) as well as in our regular PNs. These SCN processors are just as able to run the Oracle RDBMS as the PNs, and it is conceivable that in some systems there might be only SCNs and no PNs. In that case, it would probably make sense to configure the disk system with 4 SCNs per backplane, with 6 disks per SCN.

Now we're going to consider various machine configurations. We will consider one set of systems that are optimized for OLTP and another set for decision support. For the OLTP system, we compute the minimum number of processors that will still achieve the sustained number of disk IO's. For decision support, we compute the minimum number of processors that will keep up with the transfer

²This number was provided by Oracle. We have not yet verified it.

³This number was provided by Oracle. We have not yet verified it.

⁴This should be the same as a 40Mh Viking but I am assuming that a fully loaded system will have some amount of bus contention

⁵An IBM mainframe has about 20 MIPS or the same performance as a SPARC

⁶Current transfer rates are over 2 Mbytes/sec, but I am assuming that disks spend half their time seeking

⁷These are numbers that Oracle recommends. Smaller sized blocks would increase the amount of block overhead. Larger blocks would require Oracle source modifications.

rates of all the disks in the system. In some cases, a particular machine cannot be configured to keep up with the disk system in one of those two ways.

It should be understood that the configurations analyzed below are not the only possible configurations. It is possible that a customer would want a good mix of OLTP and decision support performance but is not willing to spend the money necessary to achieve peak performance for either of them. OLTP performance is the easiest to achieve and sufficiently powerful systems already exist. We will therefore limit ourselves to considering users who want maximum decision support performance.

B.1 Disclaimer

This analysis is only a first attempt at comparing these machines. It makes all sorts of assumptions and is likely to be incorrect in a variety of ways. The pricing information, for example, was pulled out of a hat. I tried hard to make up reasonable numbers, but real facts are hard to come by and I did not put in the effort to find or derive them. I think that this analysis shows how valuable it would be if it were based on real numbers. Nevertheless, I think that the conclusions that I have drawn are more correct than not.

B.2 Description of the table entries

There is a table for each of the following machine types:

- The current CM5
- The Sun2000 with 20 processors
- An IBM mainframe
- CM5 configurations with as similar performance characteristics as possible to a Sun2000
- A CM5-prime with 40Mh Vikings and large caches
- CM5-prime configurations with as similar performance characteristics as possible to a Sun2000

For each machine type, we consider several database sizes (50, 100, 200 and 500 Gbytes). Then within each database size, we pick a specific number of SCN and PNs to optimize performance for either OLTP or decision support applications.

Remember that OLTP applications want sufficient processors to keep all the disks doing drive random access IOs as fast as they are capable. decision support applications require maximized sustained transfer rates.

The “% of transfer rate achieved” row describes the fraction of disk transfer rate that the processors are able of processing. For example, a value of 50% here means that the processors are capable of processing data at only half the rate that the disks are able to deliver it. The values in this row are never larger than 100% because none of the configurations described have enough processors to process faster than the disks can deliver.

The “% of accesses achieved” row describes the fraction of random disk IO’s that the processors are capable of requesting. For example, a value of 50% here means that the disks are only doing half as many IO’s as they are capable of doing. Values in this row are sometimes larger than 100% because some configurations have enough processors to keep all the disks doing the maximum number of disk seeks they are capable of. A value of 200% means that all the disks are doing seeks as fast as they possibly can, and if the number of processors were cut in half, the disks would still be running flat out with respect to seeks.

B.3 CM5

$$\text{Price} = (\$25k \text{ per proc} * \text{number of PN}) + (\$14k \text{ per disk} * \text{number of disks})$$

	50 Gbyte		100 Gbyte		200 Gbyte		500 Gbyte	
	OLTP	DS	OLTP	DS	OLTP	DS	OLTP	DS
List price	\$700k	\$3.0m	\$1.4m	\$6.0m	\$2.8	\$12.0m	\$7.0m	\$30m
number of processors	8 SCN	8 SCN +92 PN	16 SCN	16 SCN +184 PN	32 SCN	32 SCN +368 PN	80 SCN	80 SCN +920 PN
% of transfer rate achieved	8%	100%	8%	100%	8%	100%	8%	100%
% of accesses achieved	80%	250%	80%	250%	80%	250%	80%	250%

B.4 Sun2000

In all these cases we assume that the Sun2000 has been loaded with 20 processors, which is the maximum that can be put in a Sun2000. The price numbers are estimates.

$$\text{Price} = (\$5k \text{ per proc} * 20) + (\$2k \text{ per disk} * \text{number of disks})$$

	50 Gbyte		100 Gbyte		200 Gbyte		500 Gbyte	
	OLTP	DS	OLTP	DS	OLTP	DS	OLTP	DS
List price	\$200k	\$200k	\$300k	\$300k	\$500k	\$500k	\$1.1m	\$1.1m
% of transfer rate achieved	40%	40%	20%	20%	10%	10%	4%	4%
% of accesses achieved	400%	100%	200%	50%	100%	25%	40%	10%

B.5 CM5 with same decision support performance as Sun2000

The CM5 configurations below aim to have the same decision support performance as a 20 processor Sun2000. decision support applications only care about the achieved disk transfer rate, so these configurations have the same values for this line as the Sun2000 above.

	50 Gbyte		100 Gbyte		200 Gbyte		500 Gbyte	
	OLTP	DS	OLTP	DS	OLTP	DS	OLTP	DS
List price	\$1.5m	\$1.5m	\$2.0m	\$2.0m	\$3.0m	\$3.0m	\$7.0m	\$7.0m
number of processors	8 SCN +32 PN	8 SCN +32 PN	16 SCN +24 PN	16 SCN +24 PN	32 SCN +8 PN	32 SCN +8 PN	80 SCN	80 SCN
% of transfer rate achieved	40%	40%	20%	20%	10%	10%	8%	8%
% of accesses achieved	400%	100%	200%	50%	100%	25%	80%	20%

B.6 IBM Mainframe

For this case, we assume that the mainframe has 6 CPU's, which is the maximum that can be put in an IBM mainframe. We should recognize that Oracle is not the software that an IBM owner would normally run on their mainframe. Existing database systems are probably significantly more efficient than Oracle but have the drawbacks of low level software such as assembly level programming. IBM users are migrating to DB2 which probably has the same efficiency as Oracle.

I really don't know the price of a big mainframe, but everybody tells me that they price around \$20m.

	50 Gbyte		100 Gbyte		200 Gbyte		500 Gbyte	
	OLTP	DS	OLTP	DS	OLTP	DS	OLTP	DS
List price	\$20m	\$20m	\$20m	\$20m	\$20m	\$20m	\$20m	\$20m
% of transfer rate achieved	12%	12%	6%	6%	3%	3%	1%	1%
% of accesses achieved	100%	30%	60%	15%	30%	7%	12%	3%

B.7 NCR/Teradata

All we know about the NCR machine is that is being built out of a network on Intel Pentium chips. These chips have about the same MIPS as the Viking. NCR will be direct competition for us when they figure out how to put enough of these chips in a box in a scalable manner.

B.8 Teradata (today)

I have no performance figures for the machine that Teradata is selling today.

B.9 KSR1

I do not have any specific numbers for the KSR1 at this point in time. However I believe that their CPU's have about the same MIPS as processors, and that the price of their machines are comparable to ours and so the CM5 table should probably apply to the KSR1. However KSR has yet to demonstrate more than 64 processors running together for an extended period of time, so it

is not clear whether a comparable system can be built. KSR has also had some significant problems with availability that are the result of their machine design.

B.10 nCube

I do not have any specific numbers for the KSR1 at this point in time. However I believe that their CPU's have about half the MIPS of our Cypress processors. nCube has demonstrated machines with up to 64 processors running Oracle, though there is no fundamental reason why they should not be able to run larger systems. nCube has had some serious problems with availability. If a single processor goes down, for any reason, then the whole machine goes down because all the processors are necessary for packet routing.

B.11 Meiko

I know little about the Meiko Computing Surface other than that it has 40Mh Vikings, the same processor we would like to have in our machine. It is not clear how large a system they can build and what its actual performance will be. However, we have no reason to believe that it will be slower than our machine. In fact, we should expect it to be faster.

B.12 CM5-prime (with Viking and large cache)

This table is for a hypothetical machine where the processors in both the PNs and the SCNs are 40Mh Vikings with large external caches (1 Mbyte). We do not have such a machine today, but it would not be hard to build one.

The reason for going with such a processor is that it's MIPS rating for the Oracle RDBMS is approximately three times that of our current hardware.

I am assuming that the price per processor is the same as the current hardware. It is actually possible that the hardware cost would decrease since we would not have DASH chips, and because the Viking chip set is cheaper.

Price = (\$25k per proc * number of PN) + (\$14k per disk * number of disks)

	50 Gbyte		100 Gbyte		200 Gbyte		500 Gbyte	
	OLTP	DS	OLTP	DS	OLTP	DS	OLTP	DS
List price	\$700k	\$1.3m	\$1.4m	\$2.6m	\$2.8	\$5.2m	\$7.0m	\$11.5m
number of processors	8 SCN	8 SCN +24 PN	16 SCN	16 SCN +48 PN	32 SCN	32 SCN +96 PN	80 SCN	80 SCN +182 PN
% of transfer rate achieved	24%	100%	24%	100%	24%	100%	24%	100%
% of accesses achieved	240%	250%	240%	250%	240%	250%	240%	250%

B.13 CM5-prime (with Viking and large cache) with same decision support performance as Sun2000

The CM5 configurations below aim to have the same decision support performance as a 20 processor Sun2000. decision support applications only care about the achieved disk transfer rate, so these configurations have the same values for this line as the Sun2000 above.

As with the table above, this is for a hypothetical machine where the processors in both the PNs and the SCNs are 40Mh Vikings with large external caches (1 Mbyte). We do not have such a machine today, but it would not be hard to build one.

Note that we are assuming that the performance of our processors in our machine will be higher than those in the Sun2000 because of bus contention. This needs to be measured, of course.

	50 Gbyte		100 Gbyte		200 Gbyte		500 Gbyte	
	OLTP	DS	OLTP	DS	OLTP	DS	OLTP	DS
List price	\$900k	\$900k	\$1.4m	\$1.4m	\$2.8m	\$2.8m	\$7.0m	\$7.0m
number of processors	8 SCN +8 PN	8 SCN +8 PN	16 SCN	16 SCN	32 SCN	32 SCN	80 SCN	80 SCN
% of transfer rate achieved	50%	50%	25%	25%	25%	25%	25%	25%
% of accesses achieved	480%	60%	240%	60%	240%	60%	240%	60%

Orange Project





